Time Series Forecasting: Obtaining Long Term Trends with Self-Organizing Maps

Geoffroy Simon, Amaury Lendasse, Marie Cottrell, Jean-Claude Fort, Michel Verleysen

Time Series Forecasting: Obtaining Long Term Trends with Self-Organizing Maps

G. Simon[a] *, A. Lendasse[b], M. Cottrell[c], J.-C. Fort[dc] and M. Verleysen[ac†]

[a]Machine Learning Group - DICE - Université catholique de Louvain
Place du Levant 3, B-1348 Louvain-la-Neuve, Belgium

[b]Helsinki University of Technology - Laboratory of Computer and Information Science
Neural Networks Research Centre
P.O. Box 5400, FIN-02015 HUT, FINLAND

[c]SAMOS-MATISSE, UMR CNRS 8595, Université Paris I - Panthéon Sorbonne
Rue de Tolbiac 90, F-75634 Paris Cedex 13, France

[d]Lab. Statistiques et Probabilités, CNRS C55830, Université Paul Sabatier Toulouse 3 Route de Narbonne 118, F-31062 Toulouse Cedex, France

Kohonen self-organisation maps are a well know classification tool, commonly used in a wide variety of problems, but with limited applications in time series forecasting context. In this paper, we propose a forecasting method specifically designed for multi-dimensional long-term trends prediction, with a double application of the Kohonen algorithm. Practical applications of the method are also presented.

1. Introduction

Time series forecasting is a problem encountered in many fields of applications, as finance (returns, stock markets), hydrology (river floods), engineering (electrical consumption), etc. Many methods designed for time series forecasting perform well (depending on the complexity of the problem) on a rather short-term horizon but are rather poor on a longer-term one. This is due to the fact that these methods are usually designed to optimize the performance at short term, their use at longer term being not optimized. Furthermore, they generally carry out the prediction of a single value while the real problem sometimes requires predicting a vector of future values in one step. For example, in the case of some a priori known periodicity, it could be interesting to predict all values for a period as a whole. But forecasting a vector requires either more complex models (with potential loss of performance for some of the vector components) or many distinct single value predicting models (with potential loss of the correlation information between the various values). Methods able to forecast a whole vector with the same precision for each of its components are thus of great interest.

While enlarging the prediction horizon is of course of primary interest for practitioners, there is of course some limit to the accuracy that can be expected for a long-term forecast. The limitation is due to the availability of the information itself, and not to possible limitations of the forecasting methods. Indeed, there is no doubt that, whatever forecasting method is used, predicting at long term (i.e. many time steps in advance) is more difficult that predicting at short term, because of the missing information in the unknown future time steps (those between the last known value and the one to predict). At some term, all prediction methods will thus fail. The purpose of the method presented in this paper is not to

*G. Simon is funded by the Belgian F.R.I.A.
†M. Verleysen is Senior Research Associate of the Belgian F.N.R.S.

enlarge the time horizon for which accurate predictions could be expected, but rather to enlarge the horizon for which we can have insights about the future evolution of the series. By insights, we mean some information of interest to the practitioner, even if it does not mean accurate predictions. For example, are there bounds on the future values ? What can we expect in average ? Are confidence intervals on future values large or narrow ?

Predicting many steps in advance could be realized in a straightforward way, by subsampling the known sequence, then using any short-term prediction method. However, in this case, the loss of information (used for the forecast) is obviously even higher, due to the lower resolution of the known sequence. Furthermore, such solution does not allow in a general way to introduce a stochastic aspect to the method, which is a key issue in the proposed method. Indeed, to get insights about the future evolution of a series through some statistics (expected mean, variance, confidence intervals, quartiles, etc.), several predictions should be made in order to extract such statistics. The predictions should differ; a stochastic prediction method is able to generate several forecasts by repeated Monte-Carlo runs. In the method presented in this paper, the stochastic character of the method results from the use of random draws on a probability law.

Another attractive aspect of the method presented in this paper is that it can be used to predict scalar values or vectors, with the same expected precision for each component in the case of vector prediction. Having at disposal a time series of values $x(t)$ with $1 \leq t \leq n$, the prediction of a vector can be defined as follows :

$$[x(t+1), \ldots, x(t+d)] = f(x(t), \ldots, x(t-p+1)) + \varepsilon_t \quad (1)$$

where d is the size of the vector to be predicted, f is the data generating process, p is the number of past values that influence the future values and ε_t is a centred noise vector. The past values are gathered in a p-dimensional vector called *regressor*.

The knowledge of n values of the time series (with $n >> p$ and $n >> d$) means that relation (1) is known for many $(n-p-d+1)$ time steps in the past. The modeling problem then becomes to estimate a function f that models correctly the time series for the whole set of past regressors.

The idea of the method is to segment the space of p-dimensional regressors. This segmentation can be seen as a way to make possible a local modeling in each segment. This part of the method is achieved using the Self-Organizing Map (SOM) [1]. The prototypes obtained for each class model locally the regressors of the corresponding class. Furthermore, in order to take into account temporal dependences in the series, deformation regressors are built. Those vectors are constructed as the differences between two consecutive regressors. The set of regressor deformations can also be segmented using the SOM. Once those two spaces are segmented and their dependences characterized, simulations can be performed. Using a kind of Monte-Carlo procedure to repeat the simulations, it is then possible to estimate the distribution of these simulations and to forecast global trends of the time series at long term.

Though we could have chosen some other classical vector quantization (VQ) method as only the clustering property is of interest here, the choice of the SOM tool to perform the segmentation of the two spaces is justified by the fact that SOM are efficient and fast compared to other VQ methods with a limited complexity [2] and that they provide an intuitive and helpful graphical representation.

In the following of this paper, we first recall some basic concepts about the SOM classification tool. Then we introduce the proposed forecasting method, the double vector quantization, for scalar time series and then for vector ones. Next we present some experimental results for both scalar and vector forecastings. A proof of the method stability is given in appendix.

2. The Kohonen Self-Organizing Maps

The Self-Organizing Maps (SOM), developed by Teuvo Kohonen in the 80's [1], has now become a well-known tool, with established properties [3], [4]. Self-Organizing Maps have been

commonly used since their first description in a wide variety of problems, as classification, feature extraction, pattern recognition and other related applications. As shown in a few previous works [5], [6], [7], [8], [9], [10], the SOM may also be used to forecast time series at short term.

The Kohonen Self-Organizing Maps (SOM) can be defined as an unsupervised classification algorithm from the artificial neural network paradigm. Any run of this algorithm results in a set, with a priori fixed size, of prototypes. Each one of those prototypes is a vector of the same dimension as the input space. Furthermore, physical neighbourhood relation links the prototypes. Due to this neighbourhood relation, we can easily graphically represent the prototypes in a 1- or 2-dimensional grid.

After the learning stage each prototype represents a subset of the initial input set in which the inputs share some similar features. Using Voronoi's terminology, the prototype corresponds to a centroid of a region or zone, each zone being one of the classes obtained by the algorithm. The SOM thus realizes a vector quantization of the input space (a Voronoi tessellation) that respects the original distribution of the inputs. Furthermore, a second property of the SOM is that the resulting prototypes are ordered according to their location in the input space. Similar vectors in the input space are associated either to the same prototype (as in classical VQ) or to two prototypes that are neighbours on the grid. This last property, known as the topology preservation, does not hold for other standard vector quantization methods like competitive learning.

The ordered prototypes of a SOM can easily be represented graphically, allowing a more intuitive interpretation: the 1- or 2-dimensional grid can be viewed as a 1- or 2-dimensional space where the inputs are projected by the SOM algorithm, even if, in fact, the inputs are rather projected on the prototypes themselves (with some interpolation if needed in the continuous case). This projection operation for some specific input is proceeded by determining the nearest prototype with respect to some distance metric (usually the Euclidian distance).

3. The double quantization method

The method described here aims to forecast long-term trends for a time series evolution. It is based on the SOM algorithm and can be divided into two stages: the characterization and the forecasting. The characterization stage can be viewed as the learning, while the forecasting can be viewed as the use of a model in a generalization procedure.

For the sake of simplicity, the method is first presented for scalar time series prediction (i.e. $d = 1$ in (1)) and then detailed later on for vector forecasting. Examples of the method application to scalar and vector time series will be provided in section 4.

3.1. Method description: characterization

Though the determination of an optimal regressor in time series forecasting (at least in a nonlinear prediction case) is an interesting and open question [11], it is considered here that the optimal, or at least an adequate, regressor of the time series is known. Classically, the regressor can for example be chosen according to some statistical resampling (cross-validation, bootstrap, etc.) procedure.

As for many other time series analysis methods, conversion of the inputs into regressors leads to $n - p + 1$ vectors in a p-dimensional space, where p is the regressor size and n the number of values at our disposal in the time series. The resulting regressors are denoted:

$$x_{t-p+1}^{t} = \{x(t), x(t-1), \ldots, x(t-p+1)\}, \quad (2)$$

where $p \leq t \leq n$, and $x(t)$ is the original time series at our disposal with $1 \leq t \leq n$. In the above x_{t-p+1}^{t} notation, the subscript index denotes the first temporal value of the vector, while the superscript index denotes its last temporal value.

The obtained vectors x_{t-p+1}^{t} are then manipulated and the so-called deformations y_{t-p+1}^{t} are created according to:

$$y_{t-p+1}^{t} = x_{t-p+2}^{t+1} - x_{t-p+1}^{t}. \quad (3)$$

Note that, by definition, each y_{t-p+1}^{t} is associated to one of the x_{t-p+1}^{t}. In order to highlight this link, the same indices have been used.

Putting all y_{t-p+1}^t together in chronological order forms another time series of vectors, the deformations series in the so-called deformation space to be opposed to the original space containing the regressors x_{t-p+1}^t. Of course, there exist $n-p$ deformations of dimension p.

The SOM algorithm can then be applied to each one of these two spaces, quantizing both the original regressors x_{t-p+1}^t and the deformations y_{t-p+1}^t respectively. Note that in practice any kind of SOM map can be used, but it is assumed that one-dimensional maps (or strings) are more adequate in this context.

As a result of the vector quantization by the SOM on all x_{t-p+1}^t of the original space, n_1 p-dimensional prototypes $\bar{x}_i$ are obtained ($1 \leq i \leq n_1$). The clusters associated to $\bar{x}_i$ are denoted c_i. The second application of the SOM on all deformations y_{t-p+1}^t in the deformation space results in n_2 p-dimensional prototypes $\bar{y}_j$, $1 \leq j \leq n_2$. Similarly the associated clusters are denoted c'_j.

To perform the forecasting, more information is needed than the two sets of prototypes. We therefore compute a matrix $f(ij)$ based on the relations between the x_{t-p+1}^t and the y_{t-p+1}^t with respect to their clusters (c_i and c'_j respectively). The row f_{ij} for a fixed i and $1 \leq j \leq n_2$ is the conditional probability that y_{t-p+1}^t belongs to c'_j, given that x_{t-p+1}^t belongs to c_i. In practice, those probabilities are estimated by the empirical frequencies:

$$f_{ij} = \frac{\#\{x_{t-p+1}^t \in c_i \text{ and } y_{t-p+1}^t \in c'_j\}}{\#\{x_{t-p+1}^t \in c_i\}} \quad (4)$$

with $1 \leq i \leq n_1$, $1 \leq j \leq n_2$.

Note that, for a fixed i, elements f_{ij} ($1 \leq j \leq n_2$) sum to one; this justifies the fact that each row of the matrix is an (empirically estimated) probability law. Therefore the matrix will be called *transition matrix* in the following.

The computation of this transition matrix completes the characterization part of the method.

3.2. Method description: forecasting

Once the prototypes in the original and deformation spaces together with the transition matrix are known, we can forecast a time series evolution over a rather long-term horizon h (where horizon 1 is defined as the next value $t+1$ for time t).

The methodology for such forecasting can be described as follows. First, consider a time value $x(t)$ for some time t. The corresponding regressor is x_{t-p+1}^t. Therefore we can find the associated prototype in the original space, for example $\bar{x}_k$ (this operation is in fact equivalent to determining the class c_k of x_{t-p+1}^t in the SOM). We then look at row k in the transition matrix and randomly choose a deformation prototype $\bar{y}_l$ among the $\bar{y}_j$ according to the conditional probability distribution defined by f_{kj}, $1 \leq j \leq n_2$. The prediction for time $t+1$ is obtained according to relation (3):

$$\hat{x}_{t-p+2}^{t+1} = x_{t-p+1}^t + \bar{y}_l, \quad (5)$$

where $\hat{x}_{t-p+2}^{t+1}$ is the estimate of the true x_{t-p+2}^{t+1} given by our time series prediction model. However $\hat{x}_{t-p+2}^{t+1}$ is in fact a p-dimensional vector, with components corresponding to times from $t-p+2$ to $t+1$ (see relations (2) and (3)). As in the scalar case considered here we are only interested in a single estimate at time $t+1$, we extract the scalar prediction $\hat{x}(t+1)$ from the p-dimensional vector $\hat{x}_{t-p+2}^{t+1}$.

We can iterate the described procedure, plugging in $\hat{x}(t+1)$ for $x(t)$ in (2) to compute $\hat{x}_{t-p+3}^{t+2}$ by (5) and extracting $\hat{x}(t+2)$. We then do the same for $\hat{x}(t+3)$, $\hat{x}(t+4)$, ..., $\hat{x}(t+h)$. This ends the run of the algorithm to obtain a single simulation of the series at horizon h.

Next, as the goal of the method is not to perform a single long-term simulation, the simulations are repeated to extract trends. Therefore a Monte-Carlo procedure is used to repeat many times the whole long-term simulation procedure at horizon h, as detailed above. As part of the method (random choice of the deformation according to the conditional probability distributions given by the rows of the transition matrix) is stochastic, repeating the procedure leads to different simulations. Observing those evolutions allows estimating the simulation distribution and infer global trends of the time series, as the evolution of its mean, its variance, confidence intervals, etc.

It should be emphasized once again that the double quantization method is not designed to

determine a precise estimate for time $t+1$ but is more specifically devoted to the problem of longterm evolution, which can only be obtained in terms of trends.

3.3. Generalisation: vector forecasting

Suppose that it is expected to predict vectors x_{t+1}^{t+d} of future values of the times series $x(t)$; x_{t+1}^{t+d} is a vector defined as:

$$x_{t+1}^{t+d} = \{x(t+d), \ldots, x(t+2), x(t+1)\}, \quad (6)$$

where d is determined according to a priori knowledge about the series. For example when forecasting an electrical consumption, it could be advantageous to predict all hourly values for one day in a single step instead of predicting iteratively each value separately.

As above regressors of this kind of time series can be constructed according to:

$$x_{t-p+1}^{t} = \{x_{t-d+1}^{t}, x_{t-2d+1}^{t-d}, \ldots, x_{t-p+1}^{t-p+d}\}, \quad (7)$$

where p, for the sake of simplicity, is supposed to be a multiple of d though this is not compulsory. The regressor x_{t-p+1}^{t} is thus constructed as the concatenation of d-dimensional vectors from the past of the time series, as it is the concatenation of single past values in the scalar case. As the x_{t-p+1}^{t} regressor is composed of p/d vectors of dimension d, x_{t-p+1}^{t} is a p-dimensional vector.

Deformation can be formed here according to:

$$y_{t-p+1}^{t} = x_{t-p+d+1}^{t+d} - x_{t-p+1}^{t}. \quad (8)$$

Here again, the SOM algorithm can be applied on both spaces, classifying both the regressors x_{t-p+1}^{t} and the deformations y_{t-p+1}^{t} respectively. We then have n_1 prototypes $\bar{x}_i$ in the original space, with $1 \leq i \leq n_1$, associated to classes c_i. In the deformation space, we have n_2 prototypes $\bar{y}_j$, $1 \leq j \leq n_2$, associated to classes c'_j.

A transition matrix can be constructed as a vector generalisation of relation (4):

$$f_{ij} = \frac{\#\{x_{t-p+1}^{t} \in c_i \text{ and } y_{t-p+1}^{t} \in c'_j\}}{\#\{x_{t-p+1}^{t} \in c_i\}} \quad (9)$$

with $1 \leq i \leq n_1$, $1 \leq j \leq n_2$.

The simulation forecasting procedure can also be generalised:

- consider the vector input x_{t-d+1}^{t} for time t. The corresponding regressor is x_{t-p+1}^{t};
- find the corresponding prototype $\bar{x}_k$;
- choose a deformation prototype $\bar{y}_l$ among the $\bar{y}_j$ according to the conditional distribution given by elements f_{kj} of row k;
- forecast $\hat{x}_{t-p+d+1}^{t+d}$ as

$$\hat{x}_{t-p+d+1}^{t+d} = x_{t-p+1}^{t} + \bar{y}_l; \quad (10)$$

- extract the vector

$$\{\hat{x}(t+1), \hat{x}(t+2), \ldots, \hat{x}(t+d)\}$$

from the d first columns of $\hat{x}_{t-p+d+1}^{t+d}$;

- repeat until horizon h.

For this vector case too, a Monte-Carlo procedure is used to repeat many times the whole longterm simulation procedure at horizon h. Then the simulation distribution and its statistics can be observed. This information gives trends for the long term of the time series.

Note that using the SOM to quantize the vectors x_{t-p+1}^{t} and y_{t-p+1}^{t}, the method reaches the goal of forecasting vectors with the same precision for each of their components. Indeed each component from regressors x_{t-p+1}^{t} and y_{t-p+1}^{t} has the same relative weight while the distance between the considered regressor and prototype is computed in the SOM algorithm. None of the x_{t-p+1}^{t} or y_{t-p+1}^{t} components have thus a greater importance in the modification of the prototype weight during the learning of the SOM.

3.4. Extensions

Two important comments must be done.

First, as illustrated in both examples below, it is not mandatory (in equations (1), (2), (6), (7)) to consider all successive values in the regressor; according to the knowledge of the series or to some validation procedure, it might be interesting to select regressors with adequate, but not necessarily successive, scalar values or vectors in the past.

Secondly, the vector case has been illustrated in the previous section on temporal vectors (see

equation (6)). An immediate extension of the method would be to consider spatial vectors, for example when several series must be predicted simultaneously. The equations in the previous section should be modified, but the principle of the method remains valid.

3.5. Method stability

The predictions obtained by the model described in the previous subsections should ideally be confined in the initial space defined by the learning data set. In that case, the series of predicted values y_{t-p+1}^{t} is said to be stable. Otherwise, if the series tends to infinity or otherwise diverges, it is said to be unstable. The method has been proven to be stable according to this definition; a proof is given in appendix.

4. Experimental results

This section is devoted to the application of the method on two times series. The first one is the well-known Santa Fe A benchmark presented in [12]; it is a scalar time series. The second time series is the Polish electrical consumption from 1989 to 1996 [6]. This real-world problem requires the prediction of a vector of 24 hourly values.

4.1. Methodology

In the method description, the numbers n_1 and n_2 of prototypes have not been fixed. Indeed, the problem is that different values of n_1 (n_2) result in different segmentations in the original (deformation) space and in different conditional distribution in the transition matrix. The model may thus slightly vary.

Selecting the best values for n_1 and n_2 is an important question too. Traditionally, such hyperparameters are estimated by model selection procedures such as AIC, BIC or computationally-costly resampling techniques (Leave-One-Out, k-fold cross validation, bootstrap). As it will be shown further in this paper, exact values of n_1 and n_2 are not necessary, as the sensitivity of the method around the optimums is low. A simple validation is then used to choose adequate values for n_1 and n_2. For that purpose the available data are divided into three subsets: the learning, the validation and the test set. The learning set is used to fix the values of the model parameters, such as the weights of the prototypes in the SOM and the transition matrix. The validation set is used to fix meta-parameters, such as the numbers n_1 and n_2 of prototypes in the SOM maps. The validation set is thus used for model selection. The test set aims to see how the model behaves on unused data that mimic real conditions.

The selection of n_1 and n_2 is done with regards to an error criterion, in our case a sum of squared error criterion, computed over the validation set VS:

$$e_{SSE} = \sum_{y(t+1)\in VS} (y(t+1) - \hat{y}(t+1))^2. \quad (11)$$

Once n_1 and n_2 have been chosen, a new learning is done with a new learning set obtained from the reassembled learning and validation sets. This new learning is only performed once with optimal values for n_1 and n_2.

Note that, hopefully, the sensitivity of the method to specific values of n_1 and n_2 is not high. This has been experimentally verified in all our simulations, and will be illustrated on the first example (Santa Fe A) in section 4.2.

Another crucial question is the sensitivity of the method to various runs of the SOM algorithm (with the same n_1 and n_2 values). Indeed it is well known that initial conditions largely influence the exact final result of the SOM algorithm (by final result it is meant the prototype locations, and their neighborhood relations) [13]. Nevertheless, as mentioned above, the neighborhood relations of the SOM are used for visualization purposes only; they do not influence the results of the forecast. Moreover, the location of the centroids are used to quantize the space (therefore allowing the estimation of the empirical conditional frequencies of the clusters); small variations in the centroid location have thus a low influence on each prediction generated by the method, and an even lower one on the statistics (mean, confidence intervals, etc.) estimated from the predictions. This last result has been confirmed experimentally in all our simulations, for which no significant difference was observed after different runs of the two SOM algorithms.

4.2. Scalar forecasting: Santa Fe A

The Santa Fe A time series [12] has been obtained from a far-infrared-laser in a chaotic state. This time series has become a well-known benchmark in time series prediction since the Santa Fe competition in 1991. The completed data set contains 10 000 data. This set has been divided here as follows: the learning set contains 6000 data, the validation set 2000 data, and test set 100 data. Note that the best neural network models described in [12] do not predict much more than 40 data, making a 100-data test set a *very* long-term forecasting.

Here, the regressors x_{t-p+1}^{t} have been constructed according to

$$x_{t-p+1}^{t} = \{x(t), x(t-1), x(t-2), x(t-3), x(t-5), x(t-6)\} \quad (12)$$

This choice is made according to previous experience on this series [12]. In other words, $d = 1$, $p = 6$ (as value $x(t-4)$ is omitted) and $h = 100$.

In this simulation, Kohonen strings of 1 up to 200 prototypes in each space have been used. All the 40 000 possible models have been tested on the validation set. The best model among them has 179 prototypes in the regressor space and 161 prototypes in the deformation space. After relearning this model on both the learning and validation sets, 1000 simulations were performed on a horizon of 100. Then the mean and confidence interval at 95% level were computed, giving information on the time series trends. Figure 1 shows the mean of the 1000 simulations compared to the true values contained in the test set, together with the confidence interval at 95% level. Figure 2 shows a zoom on the first 30 values. In figure 3, we can see 100 simulations for the same 30 values. Note the stability obtained through the replications. For a simpler model with $n_1 = 6$ and $n_2 = 8$ (used for illustrations purposes), figure 4 shows the code vectors and regressors (resp. deformations) in each class; table 1 shows the corresponding transition matrix.

From figure 2, it should be noted that the method gives roughly the first 25 values of the time series, a result that is not so far from those obtained with the best neural network models of the Santa Fe competition [12].

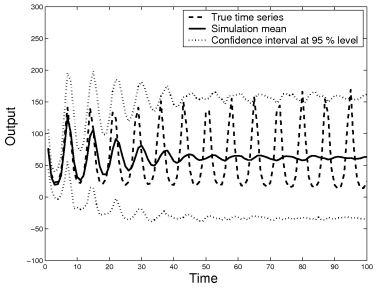


Figure 1. Comparison between the mean of the 1000 simulations (solid) and the true values (dashed), together with confidence intervals at 95% level (dotted).

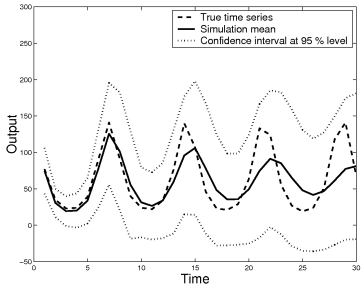


Figure 2. Comparison for the first 30 values between the mean of the 1000 simulations (solid) and the true values of the test set (dashed), together with confidence intervals at 95% level (dotted).

From figure 1, we can infer that the series mean will neither increase nor decrease. In addition, the confidence interval does contain the whole evolution of the time series for the considered 100 future values. The trend for long term forecasting is thus that the series, though chaotic, will show

some kind of stability in its evolution for the next 100 values.

As all the 40 000 models have been generated and learned, the influence of varying the n_1 and n_2 values can be observed. This influence is illustrated in figure 5. It is clear from this figure that there is a large flat region around the optimal values; in this region, all models generalize rather equivalently. This justifies, a posteriori, the choice of a simple resampling method to choose n_1 and n_2.

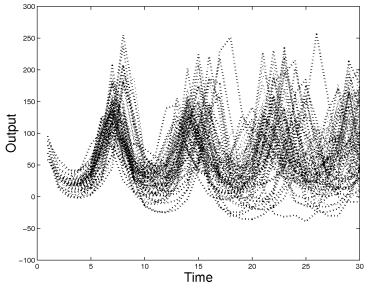


Figure 3. 100 simulations picked out at random from the 1000 simulations made for the Santa Fe A long-term forecasting.

0.12	0	0	0	0	0	0.23	0.66
0.67	0.30	0	0	0	0	0.02	0.01
0.05	0.55	0.40	0	0	0	0	0
0.03	0	0.30	0.54	0.13	0	0	0
0	0	0	0	0.50	0.48	0.02	0
0.06	0	0	0	0	0.34	0.56	0.04

Table 1
Example of transition matrix, here with $n_1 = 6$ and $n_2 = 8$ as in figure 4. Note that in each row, the frequency values sum to one.

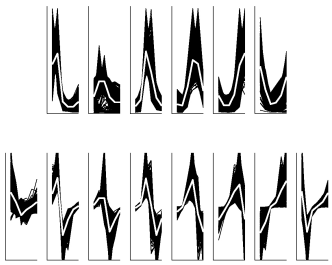


Figure 4. The code vectors and associated curves in the regressor (top) and deformation (bottom) spaces (when $n_1 = 6$ and $n_2 = 8$). The code vectors are represented in white as 6-dimensional vectors (according to (12)). Regressors (resp. deformations) belonging to each class are shown in black.

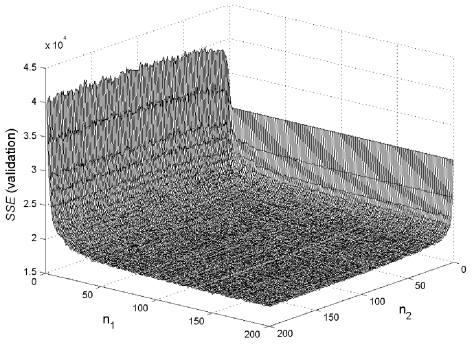


Figure 5. Impact of the variation of n_1 and n_2 on the model generalization ability for the Santa Fe A time series.

4.3. Vector forecasting: the Polish electrical consumption

As second example, we use the Polish electrical load time series [6]. This series contains hourly values from 1989 to 1996. The whole dataset con-

tains about 72 000 hourly data and is plotted in figure 6. Due to the daily periodicity of the time series, we are interested in daily predictions. This is thus an illustration of the case $d > 1$, since it seems natural to forecast the 24 next values in one step (the next day), the time window becoming daily instead of hourly.

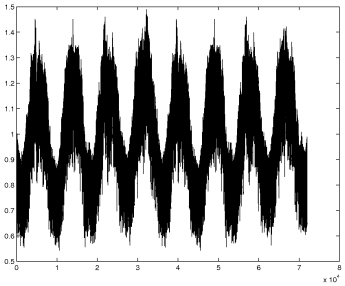


Figure 6. The Polish electrical consumption time series, between 1989 and 1996.

Having now at our disposal 3000 x_{t-p+1}^{t} data of dimension 24, we use 2000 of them for the learning, 800 for a simple validation and 200 for the test. Since the optimal regressor is unknown, many different regressors were tried, using intuitive understanding of the process. The final regressor is:

$$x_{t-p+1}^{t} = \{x_{t-24+1}^{t}, x_{t-48+1}^{t-24}, x_{t-72+1}^{t-48}, x_{t-168+1}^{t-144}, x_{t-192+1}^{t-168}\}, \quad (13)$$

that is the 24 hourly values of today, of yesterday, of two, six and seven days ago. This regressor is maybe not the optimal one, but it is the one that makes the lowest error on the validation set in comparison with other tested ones. Since the regressor contains $p = 5$ data of dimension $d = 24$, we work in a 120-dimensional space. We then run the algorithm again on the learning set with values for n_1 and n_2 each varying from 5 to 200 prototypes by steps of 5. The lowest error is made by a model with $n_1 = 160$ and $n_2 = 140$ respectively.

Another model is then learned with 160 and 140 parameter vectors in each space with the new learning set, now containing 2000+800 data. The forecasting obtained from this model is repeated 1000 times. Figure 7 presents the mean of the 1000 simulations obtained with 24-dimensional vectors and with horizon h limited to 40 days (a single plot of the whole 24 * 200 predicted values becomes unreadable). For convenience, figure 8 shows a zoom and a comparison between the mean of those 1000 long-term predictions and the real values. A confidence interval at 95% level is also provided.

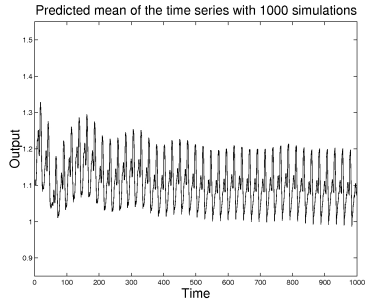


Figure 7. Mean of the 1000 simulations at long term ($h = 40$).

From figure 8, it is clear that the mean of the prediction at long term will show the same periodicity as the true time series and that the values will be contained in a rather narrow confidence interval. This fact denotes a probable low variation of the series at long term.

Figure 9 shows 100 predictions obtained by the Monte-Carlo procedure picked up at random before taking the mean. See that different simulations have about the same shape; this is a main argument for determining long-term trends.

Finally, as in the previous example, the influence of n_1 and n_2 can be observed. In figure 10,

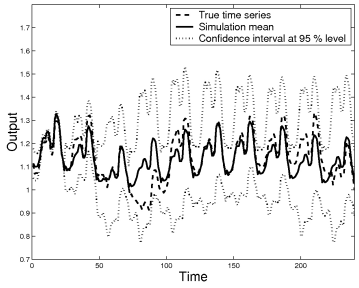


Figure 8. Comparison between the true values (dashed), the mean of the predictions (solid) and the confidence interval at 95 % level (dotted).

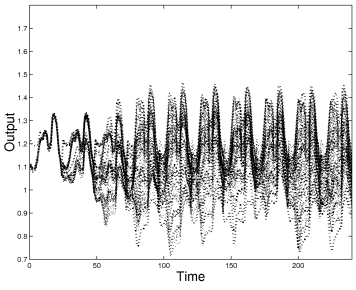


Figure 9. Plot of 100 simulations chosen at random from the 1000 simulations.

a very large flat region is also present around the best model. Sub-optimal selection of the n_1 and n_2 values will thus not penalize too heavily the model generalization abilities.

5. Conclusion

In this paper, we have presented a time series forecasting method based on a double classification of the regressors and of their deformations using the SOM algorithm. The use of SOMs

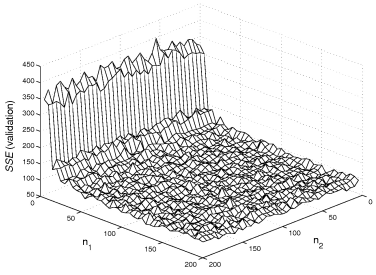


Figure 10. Impact of the variation of n_1 and n_2 on the model generalization ability for the Polish electrical consumption problem.

makes it possible to apply the method both on scalar and vector time series, as discussed in section 3 and illustrated in section 4. A proof of the method stability is given in appendix.

The proposed method is not designed to obtain an accurate forecast of the next values of a series, but rather aims to determine long-term trends. Indeed, its stochastic nature allows repeating simulations by a Monte-Carlo procedure, allowing to compute statistics (variance, confidence intervals, etc.) on the predictions. Such a method could also be used for example in the financial context, for the estimation of volatilities.

Acknowledgements

We would like to thank Professor Osowsky from Warsaw Technical University for providing us the Polish Electrical Consumption data used in our example.

REFERENCES

1. T. Kohonen, Self-organising Maps, Springer Series in Information Sciences, Vol. 30, Springer, Berlin, 1995.
2. E. de Bodt, M. Cottrell, P. Letremy, M. Verleysen, On the use of Self-Organizing Maps to accelerate vector quantization, Neurocom-

puting, Elsevier, Vol. 56 (January 2004), pp. 187-203.
3. M. Cottrell, J.-C. Fort, G. Pagès, Theoretical aspects of the SOM algorithm, Neurocomputing, 21, p119-138, 1998.
4. M. Cottrell, E. de Bodt, M. Verleysen, Kohonen maps versus vector quantization for data analysis, European Symp. on Artificial Neural Networks, April 1997, Bruges (Belgium), D-Facto pub. (Brussels), pp. 187-193.
5. M. Cottrell, E. de Bodt, Ph. Grégoire, Simulating Interest Rate Structure Evolution on a Long Term Horizon: A Kohonen Map Application, Proceedings of Neural Networks in The Capital Markets, Californian Institute of Technology, World Scientific Ed., Pasadena, 1996.
6. M. Cottrell, B. Girard, P. Rousset, Forecasting of curves using a Kohonen classification, Journal of Forecasting, Vol. 17, pp. 429-439, 1998.
7. J. Walter, H. Ritter, K. Schulten, Non-linear prediction with self-organising maps, Proc. of IJCNN, San Diego, CA, 589-594, July 1990.
8. J. Vesanto, Using the SOM and Local Models in Time-Series Prediction, In Proceedings of Workshop on Self-Organizing Maps (WSOM'97), Espoo, Finland, pp. 209-214, 1997.
9. T. Koskela, M. Varsta, J. Heikkonen, and K. Kaski, Recurrent SOM with Local Linear Models in Time Series Prediction, European Symp. on Artificial Neural Networks, April 11 1998, Bruges (Belgium), D-Facto pub. (Brussels), pp. 167-172.
10. A. Lendasse, M. Verleysen, E. de Bodt, M. Cottrell, Ph. Grégoire, Forecasting Time-Series by Kohonen Classification, European Symp. on Artificial Neural Networks, April 1998, Bruges (Belgium), D-Facto pub. (Brussels), pp. 221-226.
11. M. Verleysen, E. de Bodt, A. Lendasse, Forecasting financial time series through intrinsic dimension estimation and non-linear data projection, in Proc. of International Workconference on Artificial and Natural Neural networks (IWANN'99), Springer-Verlag Lecture Notes in Computer Science, n 1607, pp. II596-II605, June 1999.
12. A. S. Weigend, N.A. Gershenfeld, Times Series Prediction: Forecasting the future and Understanding the Past, Addison-Wesley Publishing Company, 1994.
13. E. de Bodt, M. Cottrell, M. Verleysen, Statistical tools to assess the reliability of self-organizing maps, Neural Networks, Elsevier, Vol. 15, Nos. 8-9 (October-November 2002), pp. 967-978.
14. G. Fayolle, V. A.Malyshev, M. V. Menshikov, Topics in constructive theory of countable Markov chains, Cambridge University Press, 1995.

Appendix

Method stability

Intuitively, the stability property of the method is not surprising. Indeed, the model is designed such that it will mostly produce predictions that are in the range of the observed data. By construction, deformations are chosen randomly according to an empirical probability law and the obtained predictions should stay in the same range. If, for some reason, the prediction is about to exceed this range during one of the simulations, the next deformations will then tend to drive it back inside this range, at least with high probability. Furthermore, as simulations are repeated with the Monte-Carlo procedure, the influence of such unexpected cases will be reduced when the mean is taken to obtain the final predictions. The following of this section is intended to prove this intuitive result.

The proof consists in two steps: it is first shown that the series generated by the model is a Markov chain; secondly, it is demonstrated that this particular type of Markov chain is stable. In order to improve the readability of the proof, lighter notations will be used. For a fixed d and a fixed p, notation X_t will represent the vector x_{t-p+1}^t. The last known regressor will be denoted X_0. The prototype of a cluster C_j' of deformations will be noted Y_j . Finally, hats will be omitted for simplicity as all regressors X_t are estimations, except for $t = 0$.

To prove that the series is a Markov chain, we consider the starting vector of the simulation at time 0. The corresponding initial regressor of the series is denoted X_0, and C_0 is the corresponding SOM cluster in the regressor space. The deformation that is applied to X_0 at this stage is Y_0. Then the next values of the series are given by $X_1 = X_0 + Y_0$, $X_2 = X_0 + Y_0 + Y_1$, ..., with Y_0, Y_1, ... drawn randomly from the transition matrix for clusters C_0, C_1, ... respectively. The series X_t is therefore a Markov chain, homogeneous in time (the transition distribution are not time dependant), irreducible and defined over a numerable set (the initial X_t are in finite number, and so are the deformations).

To show the stability of this Markov chain and thus the existence of a stationary distribution, Foster's criterion [14] is applied. Note that this criterion is a stronger result which proves the ergodicity of the chain, which in turns implies the stability. Foster's criterion is the following:

A necessary and sufficient condition for an irreducible chain to be ergodic is that there exists a positive function $g(.)$, a positive ε and a finite set A such that:

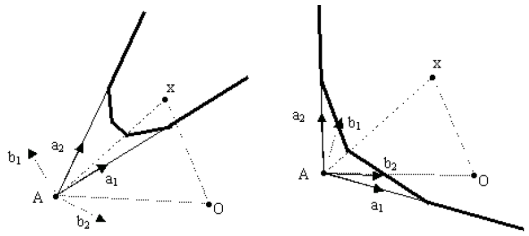
$$\begin{aligned} &\forall x \in \Omega : E(g(X_{t+1})|X_t = x) < \infty, \\ &\forall x \notin \Omega : E(g(X_{t+1})|X_t = x) - g(x) \leq -\varepsilon. \end{aligned} \tag{14}$$

Since the Markov chain is homogenous, it is sufficient to observe transition Y_0 from X_0 to X_1. The same development can be deduced for any other transition.

The demonstration is done for two-dimensional regressors but can be generalized easily to other dimensions. Note that in the following, we use $g(.) = \|.\|^2$ in (14).

Before going in further details, let us remark that for a SOM with at least 3 classes in general position, class C_0 covers less than a half plane. Furthermore, we have to distinguish two cases for each cluster. First, the cluster may be included in a finite compact from $\mathbb{R}^2$. The second case is the case of an infinite cluster i.e. of a cluster which may does have any neighbour in some direction; this happens to ckusters on the border of the map.

The first case is easely proved. Since $\|X_0\| <$



(a) A Cluster within an acute angle

(b) A Cluster within an obtuse angle

Figure 11. Notations for the cone containing an unbounded cluster of a SOM; see text for details.

R_0, where R_0 can be any constant, then we have by triangular inequality:

$$\begin{aligned} E(\|X_1\|) &< R_0 + \|Y_0\| \\ &\leq R_0 + \max_j(\|Y_j\|). \end{aligned} \tag{15}$$

As the deformations Y_j are in finite number, the maximum of their norm is finite. This proves the first inequality of (14) in an obvious way for the first case (i.e. bounded cluster case).

The other case thus appens when $\|X_0\| \to +\infty$. This happens in unbounded clusters. The unbounded cluster case is much more technical to prove.

Looking at figure 11, we see that each unbounded cluster is included in a cone with vertex A and delimited by the normalized vectors a_1 and a_2. There are two possibilities: either a_1 and a_2 form an acute angle, either an obtuse one, as shown in figure 11(a) and figure 11(b) respectively.

Before going on and applying Foster's criterion, note that the three following geometrical properties can be proven:

Property 1.
Denoting

$$\lim_{\|x\| \to \infty} \frac{x}{\|x\|} \cdot a_i = \delta_i, \tag{16}$$

we have δ_1 and δ_2 both positive in the acute angle case, while either δ_1 or δ_2 is positive for an obtuse angle. Indeed, using the origin O, we define:

$$\overrightarrow{Ox} = \overrightarrow{OA} + \overrightarrow{Ax}. \tag{17}$$

We thus have:

$$\frac{x}{\|x\|} \cdot a_i = \frac{\overrightarrow{OA} \cdot a_i}{\|x\|} + \frac{\overrightarrow{Ax}}{\|\overrightarrow{A}x\|} \frac{\|\overrightarrow{A}x\|}{\|x\|} \cdot a_i \tag{18}$$

which can be bounded by a strictly positive constant as $\frac{\overrightarrow{OA} \cdot a_i}{\|x\|} \to 0$ and $\frac{\|\overrightarrow{Ax}\|}{\|x\|} \to 1$ for $\|x\| \to +\infty$.

Property 2.

We define b_1 such that the angle (a_1, b_1) is $+\frac{\pi}{2}$. Similarly b_2 is defined such that the angle (b_2, a_2) is also $+\frac{\pi}{2}$. Then, for both the acute and obtuse angle cases, we have:

$$\inf_{x \in C} \frac{\overrightarrow{Ax}}{\|x\|} \cdot b_i = r_i > 0, \tag{19}$$

where C is the considered cone which has border vectors a_1 and a_2.

Rewrite the first term of (19) as:

$$\inf_{x \in C} \frac{\overrightarrow{Ax}}{\|x\|} \cdot b_i = \inf_{x \in C} \frac{\overrightarrow{Ax}}{\|\overrightarrow{A}x\|} \frac{\|\overrightarrow{A}x\|}{\|x\|} \cdot b_i; \tag{20}$$

the result is obtained easily since $\frac{\|\overrightarrow{A}x\|}{\|x\|} \to 1$ when $\|x\| \to +\infty$.

Property 3.

Assume that:

$$E_{\mu_0}(Y_0) \cdot a_1 < 0 \text{ and } \mathrm{E}_{\mu_0}(\mathrm{Y}_0) \cdot \mathrm{a}_2 < 0 \tag{21}$$

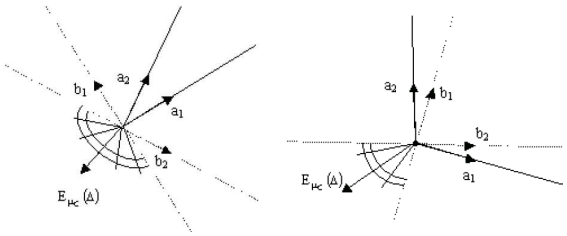
where μ_0 is the empirical distribution corresponding to class C_0 in the transition matrix. Denoting

$$E_{\mu_0}(Y_0) \cdot a_i = -\gamma_i < 0 \tag{22}$$

with $\gamma_i > 0$, then we have:

$$E_{\mu_0}(Y_0) \cdot b_i < 0 \tag{23}$$

for either $i = 1$ or $i = 2$ in case of an acute angle (figure 12(a)) or for both of $i = 1$ and $i = 2$ for the obtuse case (figure 12(b)).



(a) Acute angle case. (b) Obtuse angle case.

Figure 12. Third geometrical property, see text for details.

Note that the initial assumption can easily be proved numerically.

Those three properties will be used as lemmas in the following. Now we can apply Foster's criterion for the unbounded cluster case.

Foster's criterion

Considering an unbounded class C_0 and the corresponding transition distribution, with $g(x) = \|x\|^2$, we have

$$\begin{aligned} &E(g(X_1)|X_0 = x) - g(x) \\ &\quad = E(g(X_0 + Y_0|X_0 = x) - g(x) \\ &\quad = E(\|X_0 + Y_0\|^2|X_0 = x) - \|x\|^2 \\ &\quad = 2\|x\| \left[\frac{x \cdot E_{\mu_0}(Y_0)}{\|x\|} + \frac{E_{\mu_0}(\|Y_0\|^2)}{2\|x\|} \right]. \end{aligned} \tag{24}$$

The second term between the brackets can be bounded by a strictly positive constant α_0. Indeed, as $\|Y_0\|^2$ is finite, $E_{\mu_0}(\|Y_0\|^2) < M_0$ is also finite. Therefore, for $\alpha_0 > 0$ and $\|x\| > \frac{M_0}{\alpha_0}$, we have

$$\frac{1}{\|x\|} E_{\mu_0}(\|Y_0\|^2) < \alpha_0. \tag{25}$$

For the first term, we chose either $i = 1$ or $i = 2$ such that:

$$\begin{cases} \lim\limits_{\|x\| \to +\infty} \dfrac{x}{\|x\|} \cdot a_i = \delta_i > 0, \\ E_{\mu_0}(Y_0) \cdot b_i < 0. \end{cases} \tag{26}$$

In case of an unbounded cluster, those two conditions are fulfilled using Properties 1. and 3.

By hypothesis, suppose that $i = 2$ satisfies those two conditions (26). The term $E_{\mu_0}(Y_0)$ can be decomposed in the (b_2, a_2) basis. Then, for $\|x\|$ sufficiently large, as:

- $E_{\mu_0}(Y_0) \cdot a_2 = -\gamma_2$ by Property 3.;
- $\frac{x}{\|x\|} \cdot a_2 > \frac{\delta_2}{2}$ by Property 1.;
- $E_{\mu_0}(Y_0) \cdot b_2 < 0$ by Property 3.;
- $\frac{x}{\|x\|} \cdot b_2 \geq \frac{r_2}{2}$ as $\overrightarrow{Ox} = \overrightarrow{OA} + \overrightarrow{Ax}$ and by Property 2.;

we have

$$\begin{aligned}
&\frac{x}{\|x\|} E_{\mu_0}(Y_0) \\
&\quad \leq \underbrace{(E_{\mu_0}(Y_0) \cdot a_2)}_{=-\gamma_2} \underbrace{\left(\frac{x}{\|x\|} \cdot a_2\right)}_{>\frac{\delta_2}{2}} \\
&\qquad + \underbrace{(E_{\mu_0}(Y_0) \cdot b_2)}_{<0} \underbrace{\left(\frac{x}{\|x\|} \cdot b_2\right)}_{\geq \frac{r_2}{2}} \\
&\quad < -\gamma_2 \frac{\delta_2}{2},
\end{aligned}$$

when $\|x\|$ is large enough, denoted here $\|x\| > L_0$.

The same development can be achieved using $i = 1$ to satisfy the two initial conditions (26). We obtain:

$$\frac{x}{\|x\|} E_{\mu_0}(Y_0) < -\gamma_1 \frac{\delta_1}{2}, \tag{27}$$

when $\|x\| > L_0'$.

Equation (24) can now be simplified in:

$$\begin{aligned}
&E(g(X_1)|X_0 = x) - g(x) \\
&\quad = 2\|x\| \left[\tfrac{x \cdot E_{\mu_0}(Y_0)}{\|x\|} + \tfrac{E_{\mu_0}(\|Y_0\|^2)}{2\|x\|}\right] \\
&\quad < 2\|x\| \left[-\alpha_0 + \tfrac{1}{2}\alpha_0\right] \\
&\quad = -2\|x\| \tfrac{\alpha_0}{2},
\end{aligned} \tag{28}$$

where $\|x\| > K_0 = \max(L_0, L_0')$ and α_0 in (25) is chosen such that $\alpha_0 = \min\left(\frac{\gamma_1\delta_1}{2}, \frac{\gamma_2\delta_2}{2}\right)$.

This development has been done for cluster C_0. All values α_0, M_0, L_0, K_0 depends on this cluster C_0. Now considering all unbounded clusters C_i and taking $\alpha = \inf_{C_i} \alpha_i$ and $K = \sup_{C_i} K_i$, we have:

$$\begin{aligned}
&\forall \|x\| \geq K : \\
&\quad \frac{x E_{\mu_0}(Y_0)}{\|x\|} + \frac{E_{\mu_0}(\|Y_0\|^2)}{2\|x\|} < -\frac{\alpha}{2} < 0.
\end{aligned} \tag{29}$$

Finally, we obtain, using (29) in (28):

$$E(g(X_1)|X_0 = x) - g(x) < -\alpha\|x\|, \tag{30}$$

where the right member tends to $-\infty$ for $\|x\| \to +\infty$.

To conclude, we define the set Ω used in Foster's criterion according to

$$\Omega = \left(\bigcup_{i \in I} C_i\right) \bigcup \{X_0 |\ \|X_0\| < K\}, \tag{31}$$

where I denotes the set of bounded cluster indexes as discussed in the introduction to the proof. With this definition, the above developments prove Foster's criterion (14). Thus the Markov chain defined by the X_i for $i > 0$ is ergodic, and admits a unique stationary distribution.